



Enabling Grids for E-science

PPARC Summer School,  
May 2005


## Web Services Descriptions and SOAP messages

*Richard Hopkins,  
National e-Science Centre, Edinburgh*

[www.eu-egee.org](http://www.eu-egee.org)



## Goals –

- To be able to understand
  - WSDL definition for a standard SOAP binding
  - A soap message
  
- Structure
  - SOAP Messages 
  - General Structure of WSDL
  - Details of abstract Service Definition
  - Details of Physical Service Definition - Core
  - Physical Service Definition - Extensions

- **Name**
  - Originally – Simple Object Access Protocol
  - Temporarily – Service Oriented Architecture Protocol ?
  - Now (SOAP 1.2) – Not an acronym
- **Purpose**
  - A extensible protocol to enable the exchange of
    - structured and typed information
    - between peers
    - in a decentralised, distributed environment
- **Status**
  - SOAP 1.2 – <http://www.w3.org/TR/soap12-part0>
    - W3C recommendation, June 2003
  - **SOAP 1.1** – <http://www.w3.org/TR/NOTE-SOAP-20000508>
    - W3C submission May 2000 – but that’s what people use currently

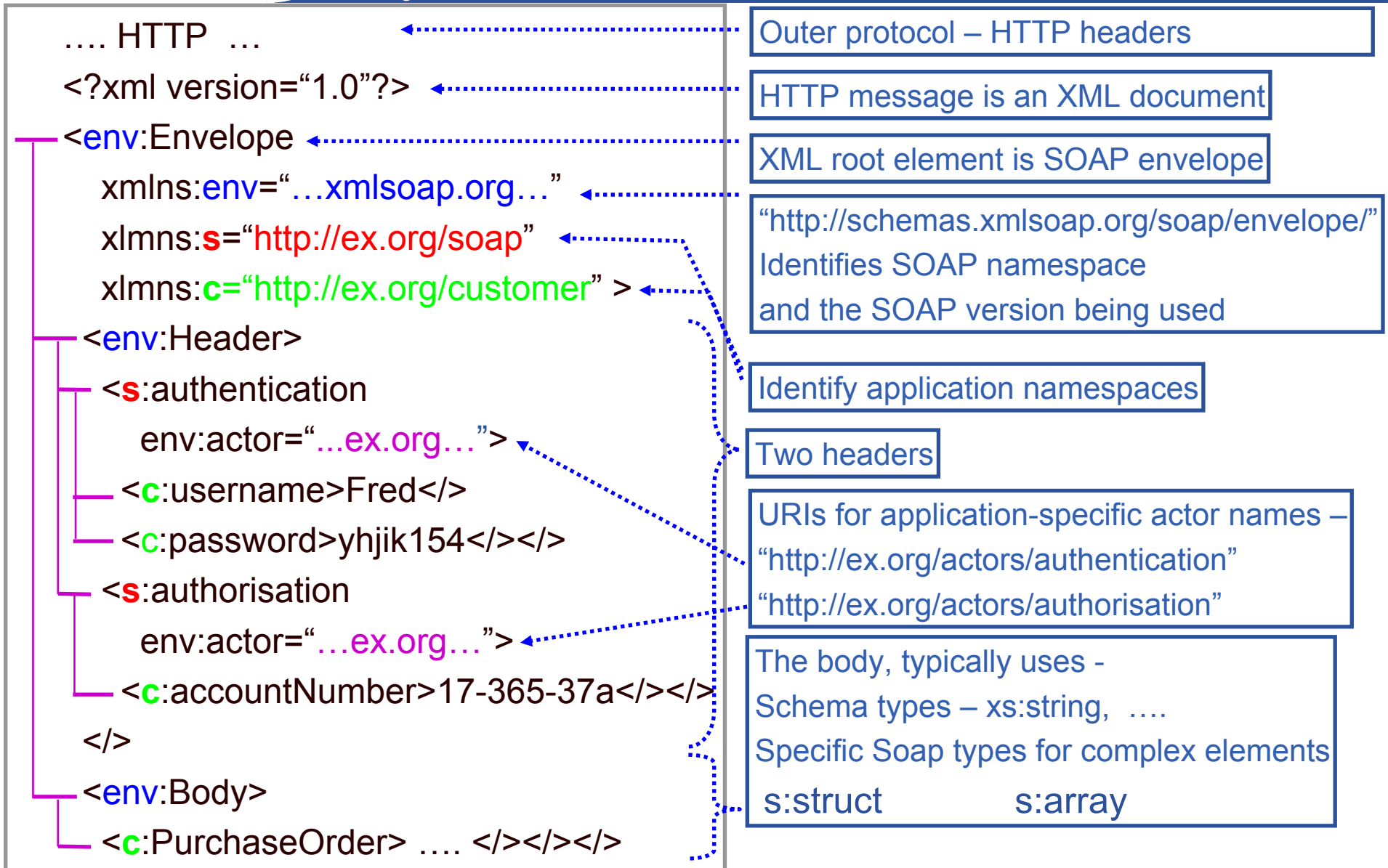
- **XML based**
  - Message is an XML document
  - Allow attachments in other (more space efficient) formats
- **Higher order Protocol –**
  - Built on some underlying protocol - binding
    - Extensibility – can define binding for any underlying protocol
    - Usually HTTP – a specific standard extension
- **Single Message Protocol**
  - Defines standard for a single message communication
  - Multi-message conversations require a means to associate one message with another
    - Via underlying protocol (e.g. use of same connection)
    - Via the application (specific message-id information as part of the soap message)
- **Multi-stage message processing –** The soap Processing model
- **Standard Error reporting –** Message format ; error codes

**Each SOAP message will have:**


- **Outer layer(s) for underlying protocols**
  - Only consider HTTP
- **Envelope (XML root element)**
- **Header (optional)**
  - Multiple header blocks/entries
  - For different purposes – factorisation
  - For different processing stages
    - Actors
- **Body (mandatory)**
  - The payload
  - Zero or more XML elements
  - May be a Fault element
    - Specific fault reporting standard



# XML Message Representation



## Goals –

- To be able to understand
  - WSDL definition for a standard SOAP binding
  - A soap message
  
- Structure
  - SOAP Messages
  - General Structure of WSDL 
  - Details of abstract Service Definition
  - Details of Physical Service Definition - Core
  - Physical Service Definition - Extensions

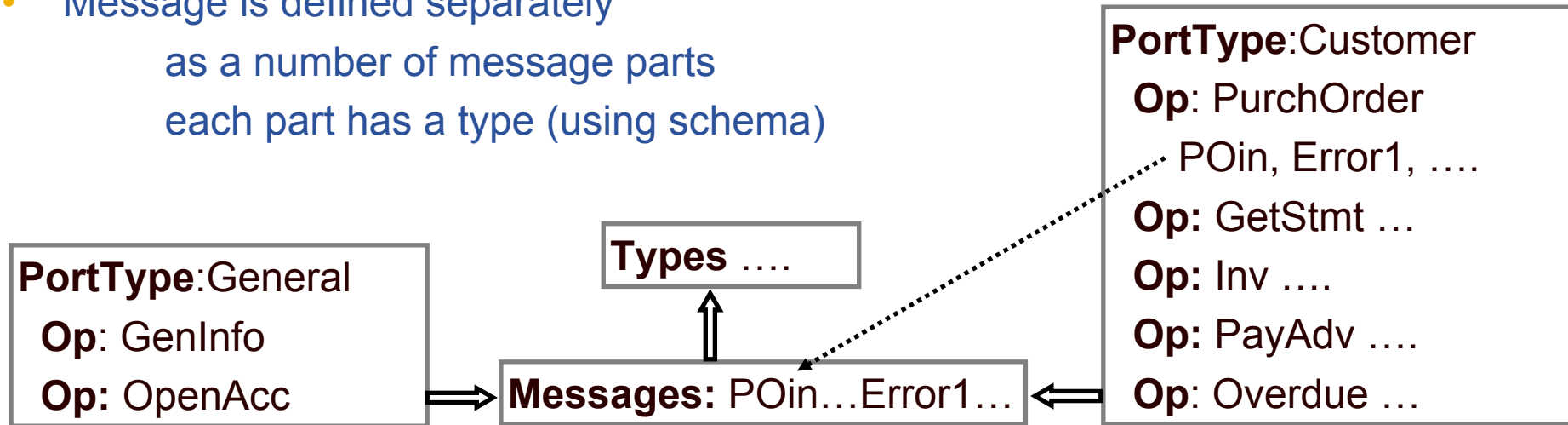
- **An XML format**
- **For describing network services**
  - Operates either on
    - Documents
    - Procedure calls
  - Describes the **exposed** interface –  
what the consumer sees of the service
  - Constitutes a contract with the client
    - Provides a specification of what is offered by the service provider which can be relied on by the service consumer
- **Supports Separation of concerns**
  - abstract structure – operations and messages
  - binding to a specific underlying protocol
  - definition of a particular deployed service
  - To allow common definition and re-combination
- **Here using WSDL 1.1 – a W3C submission (March 2001)**
  - 2.0 – is a last call working draft (Aug 2004) – many differences



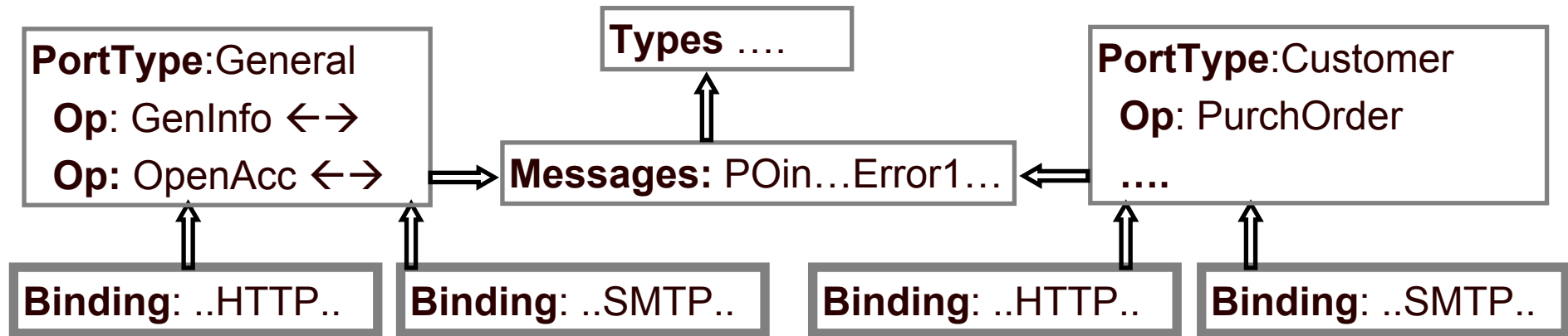
- **Example**
- **Company Provides two types of Service (PortTypes)**
  - General Service
    - Get general information (GenInfo)
    - Open an Account (OpenAcc)
  - Customers Service (being a “Customer” = having an account)
    - Purchase Order (PurchOrder)
    - Invoice (Inv)
    - Payment Advice (PayAdv)
    - Get Statement (GetStmnt)
    - Notify overdue payment (Overdue)
- **Both over two kinds of binding**
  - Web - HTTP
  - Email - SMTP

# Abstract Definition - PortTypes

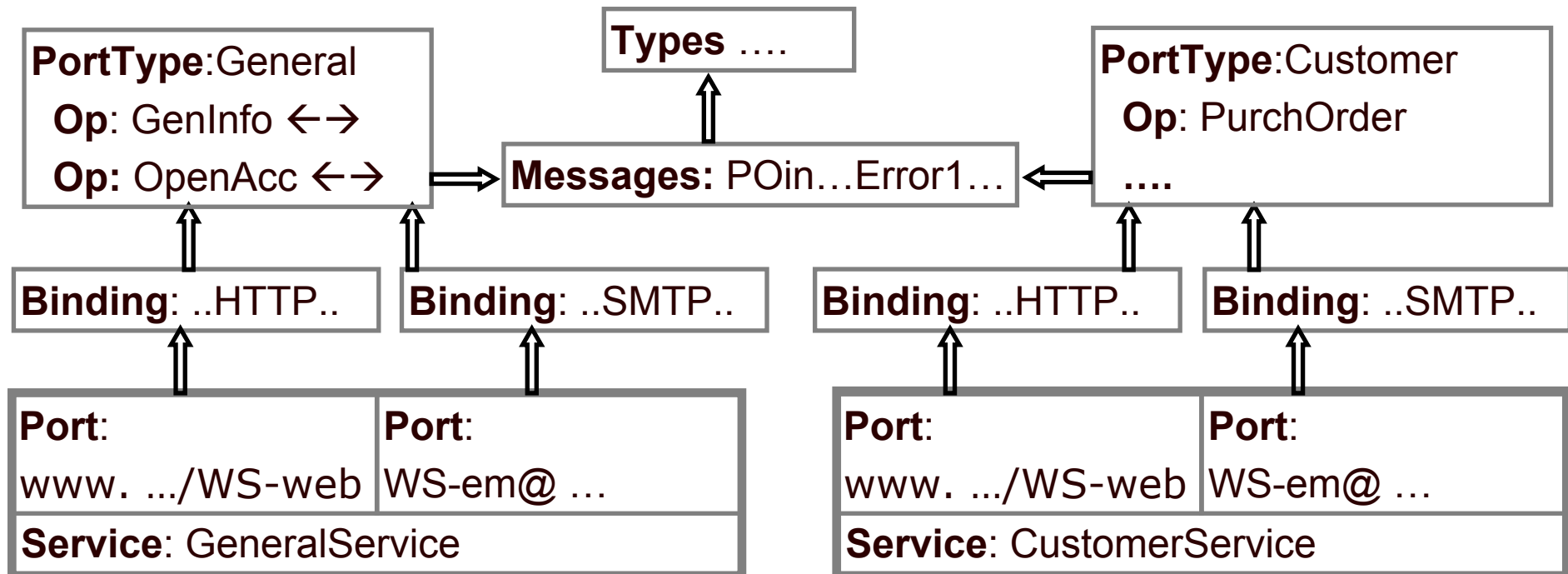
- Start with PortType = Interface
- Set of operations
- For each operation, a number of messages – input; output; faults
- Message is defined separately  
as a number of message parts  
each part has a type (using schema)



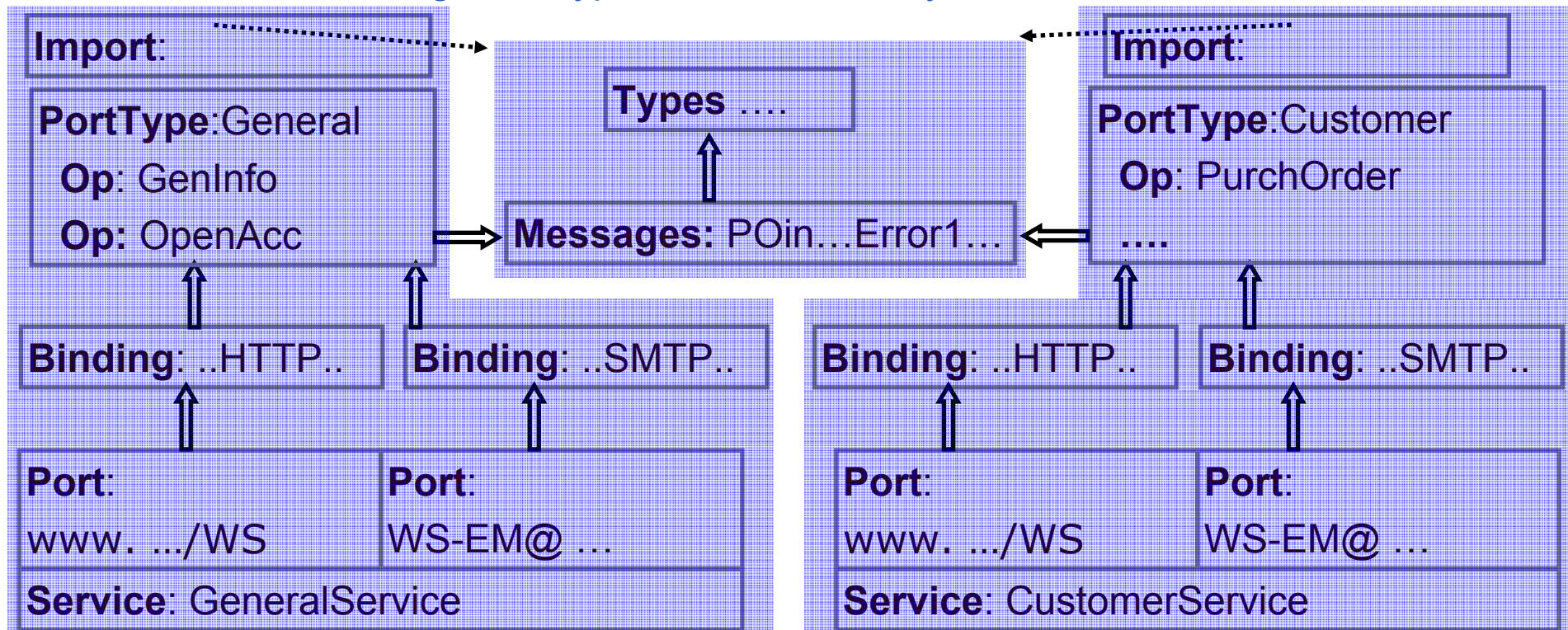
- **Binding –**
  - A binding of a portType to a communication protocol for using it
  - Specifies
    - The portType
    - The underlying protocol(s)
    - How the logical structure is represented using the underlying protocol
  - Here two bindings for each PortType – web, e-mail

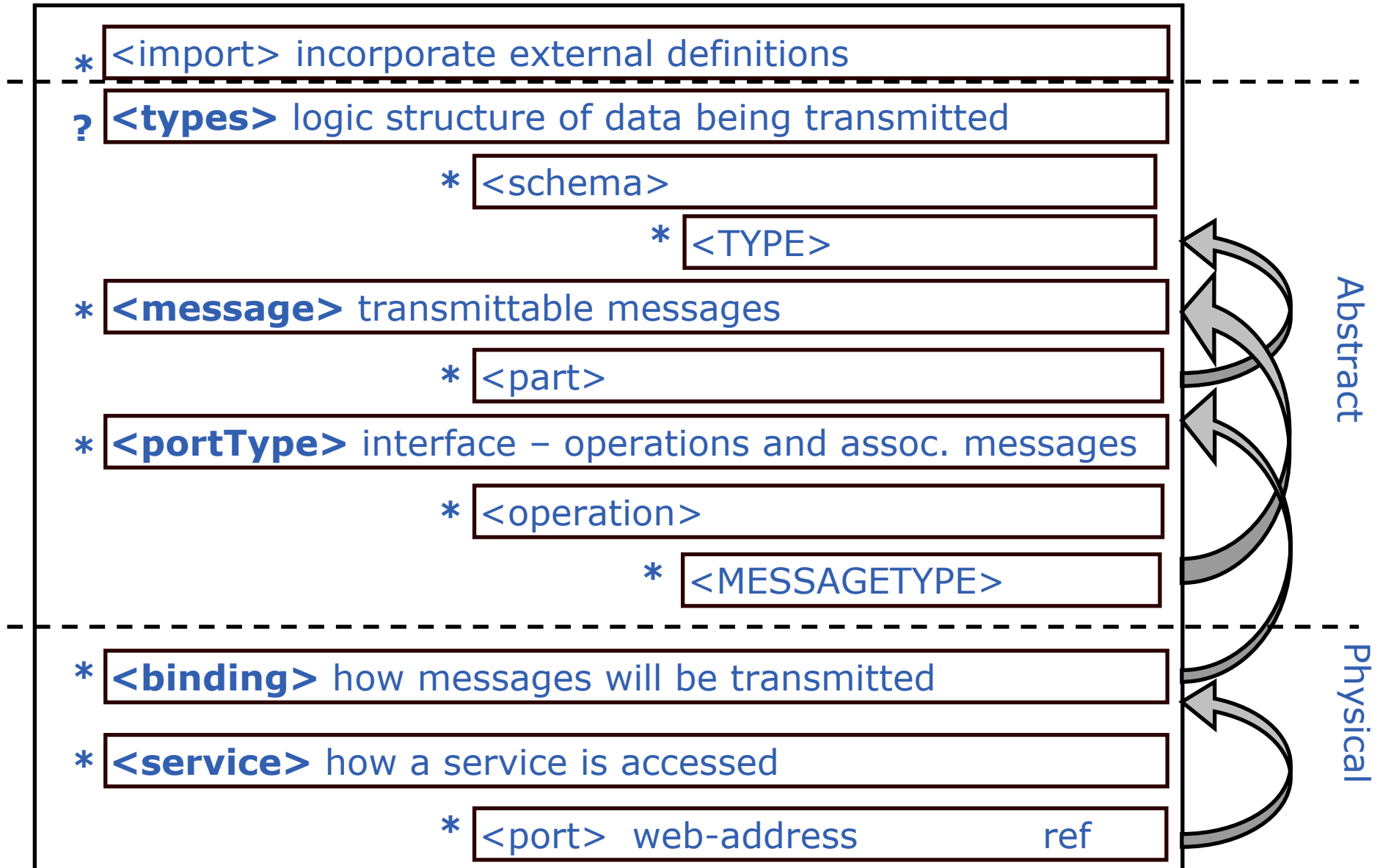


- **Service – defines one or more ports, each with**
  - Location – URL – here sharing of locations
  - Binding and thus portType
    - The interface provided by the port
    - how it is realised over a particular protocol
  - Here one service for each portType – there are alternatives ....



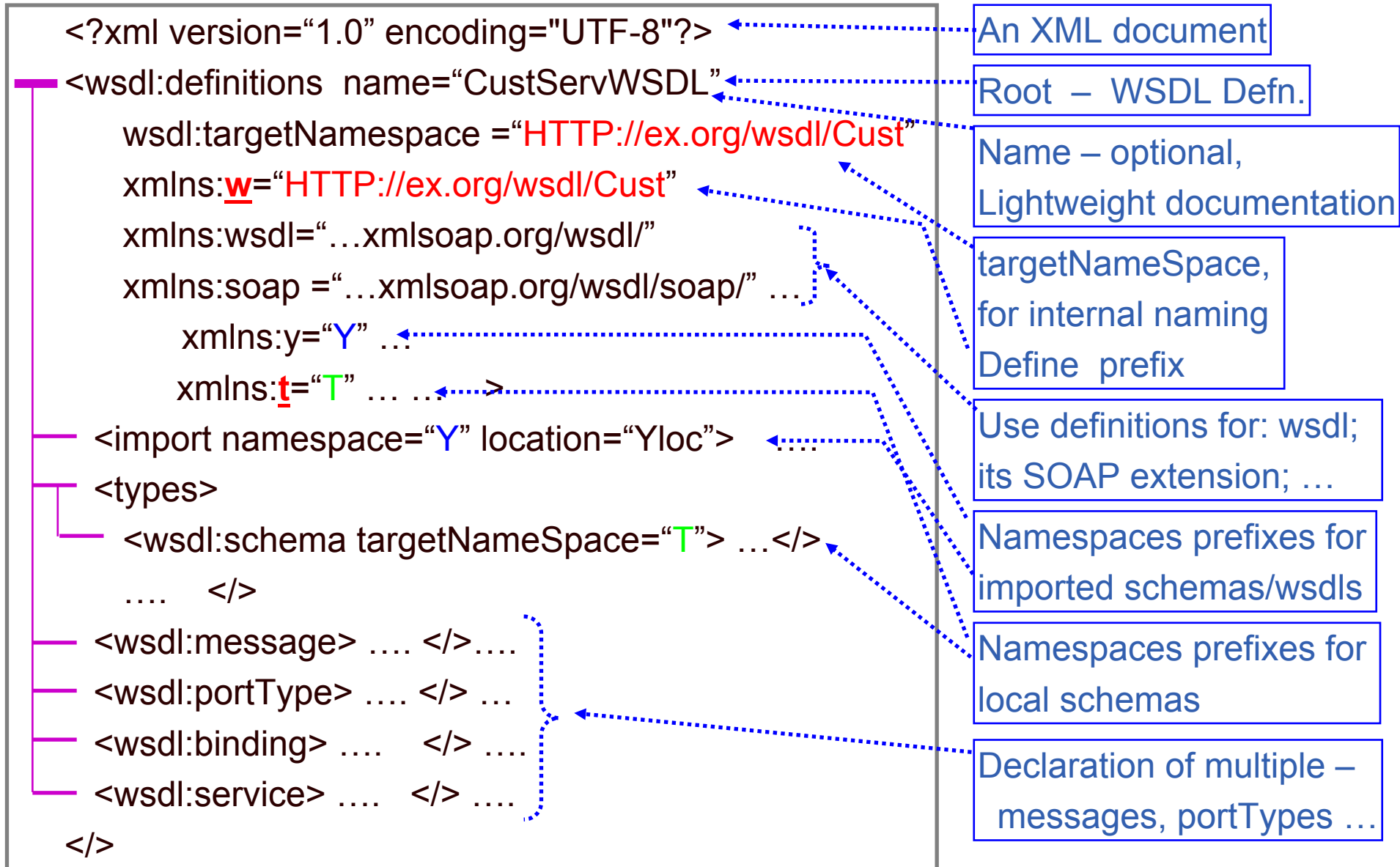
- Could put all definitions in one WSDL file – that’s what is produced by JAX
- For hand-crafted WSDL, could spread over multiple files, e.g. -
  - one WSDL file per service –
    - Gives control over publication use – CustomerService not in public registry
    - Different services may have semi-independent development
  - common message and type definitions – may be shared between interfaces






```

<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="CustServWSDL"
  wsdl:targetNamespace = "HTTP://ex.org/wsdl/Cust"
  xmlns:w="HTTP://ex.org/wsdl/Cust"
  xmlns:wsdl="...xmlsoap.org/wsdl/"
  xmlns:soap = "...xmlsoap.org/wsdl/soap/" ...
  xmlns:y="Y" ...
  xmlns:t="T" ...
  <import namespace="Y" location="Yloc">
  <types>
    <wsdl:schema targetNameSpace="T"> ...</>
    .... </>
  <wsdl:message> .... </>....
  <wsdl:portType> .... </> ...
  <wsdl:binding> .... </> ....
  <wsdl:service> .... </> ....
</>
  
```



- An XML document
- Root – WSDL Defn.
- Name – optional, Lightweight documentation
- targetNameSpace, for internal naming
- Define prefix
- Use definitions for: wsdl; its SOAP extension; ...
- Namespaces prefixes for imported schemas/wsdl
- Namespaces prefixes for local schemas
- Declaration of multiple – messages, portTypes ...

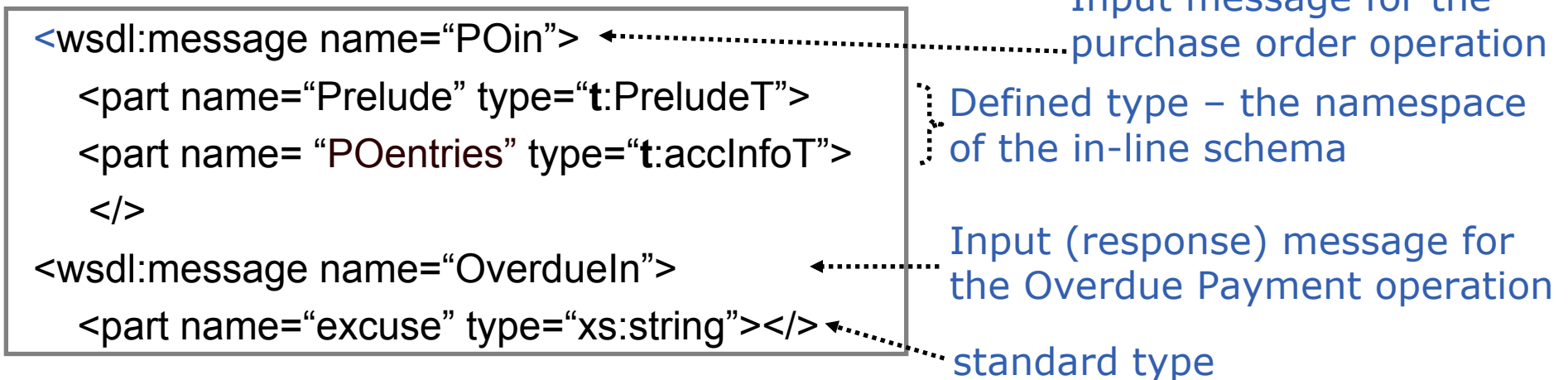
## Goals –

- To be able to understand
  - WSDL definition for a standard SOAP binding
  - A soap message
  
- Structure
  - SOAP Messages
  - General Structure of WSDL
  - Details of abstract Service Definition 
  - Details of Physical Service Definition - Core
  - Physical Service Definition - Extensions



## Message

- Has a name – so message can be referenced by a portType definition
- Consists of one or more parts,
  - Each part is a logical unit, e.g. a parameter
    - No parts in WSDL 2.0
  - Has a name so that it can be referenced by a Binding definition
    - E.g to put one part in a header and the other part in the body
  - Has a type –
    - a Schema type definition or a Schema element definition
    - a standard type – from an imported Schema



PORTTYPE – an interface comprising a set of operations

- Organisation of functionality into portTypes and operations is similar to O-O design
- A portType is a coherent unit of exposed functionality –
  - operations make sense together
    - E.g. Currency conversion might be a service used in processing customer transaction
    - But would not expect a convertCurrency operation for this service
  - provide self-contained functionality
    - Not have PO and Invoice as separate services
- But ... larger granularity than O-O
- ... Deployment considerations
  - Split between General and Customer may be that say General has a wider range of available bindings/locations
- Each operation declares a number of messages which can be communicated as the interface to the operation

```

PortType:Customer
  Op: PurchOrder
    In: POin
    Out: POout
    Fault: Error1
    ....
  Op: Overdue
    ....
  Op: Inv
    ....
  ....
  
```

```

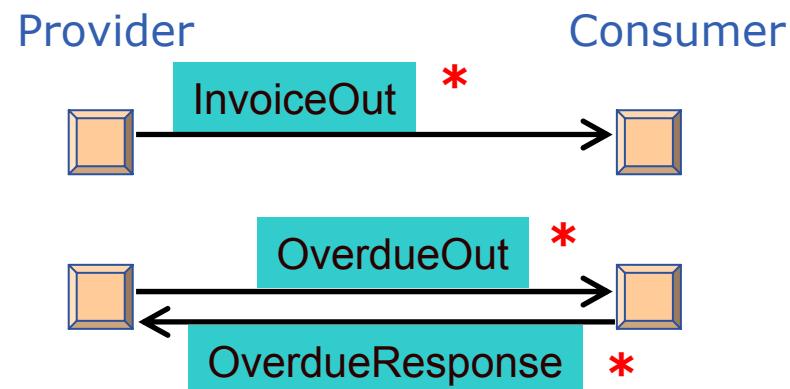
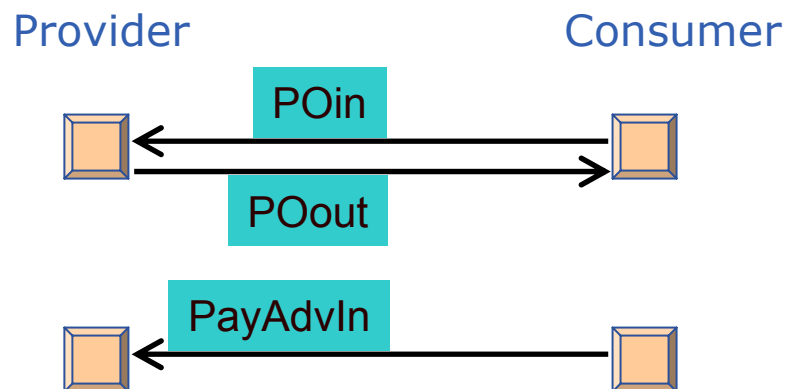
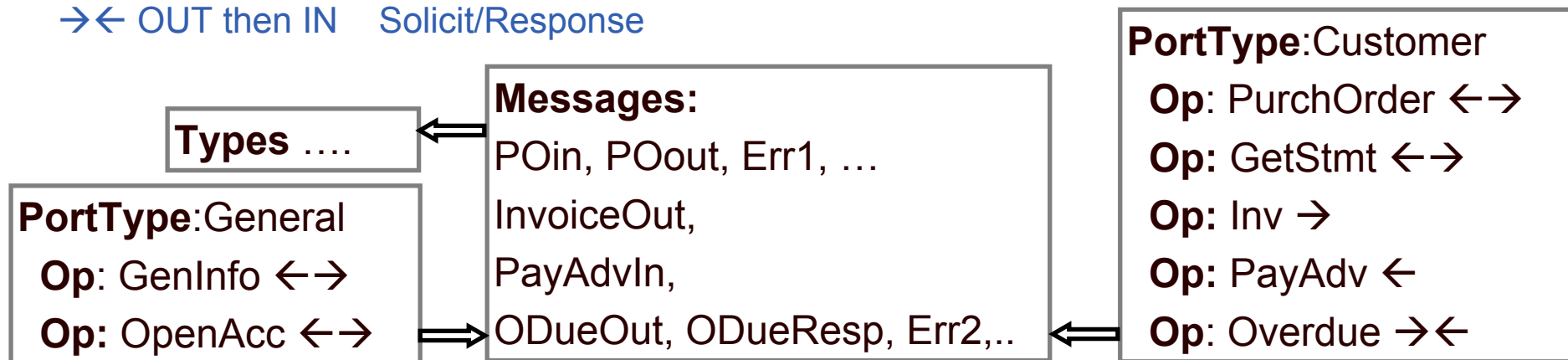
PortType:General
  Op: GenInfo
    ....
  Op: OpenAcc
    ....
  
```

# Message Exchange Patterns

Messages conform to one of Four message exchange patterns (MEPs)

- ↔ IN then OUT Request/Response – most usual
- OUT Notify
- ← IN One-way \*
- ← OUT then IN Solicit/Response

- \* Reversed roles
- Service Provider proactive = client
- Service Consumer reactive = server



- **Message exchange pattern is determined by sequence of message declarations**
  - $\leftrightarrow$  Request-Response – **input**  
**output**  
**fault\***
  - $\rightarrow\leftarrow$  Solicit-Response – **output**  
**input**  
**fault\***
  - $\rightarrow$  Notify – **output**
  - $\leftarrow$  One-way (Request) – **input**
- **Single message patterns can't have fault message**
- **(in WSDL 2.0 this is explicit and more general – named patterns)**

**PortType:Customer**  
**Op:** PurchOrder  
**In:** POin  
**Out:** POout  
**Fault:** Error1 ...  
**Op:** Overdue  
**Out:** OverdueOut  
**In:** OverdueIn  
**Fault:** ErrorThreat  
**Op:** Inv  
**Out:** InvOut  
**Op:** PayAdv  
**In:** PayAdvIn

- Message to service provider; reply to service consumer; possible fault messages
- A logical pattern, Binding might be e.g. An HTTP request/response or two HTTP requests

```

<wsdl:portType name="CustomerP">
  <wsdl:operation name="PurchOrder">
    <wsdl:input name="PurchOrderRequest"
      message="w:POin">
    <wsdl:output name="PurchOrderResponse"
      message="w:POout">
    <wsdl:fault name="Error1"
      message="w:Error1">
      ..... </>
  </wsdl:operation ...> </> ...</>
  
```

```

<wsdl:message name="POin">
  <part name="Prelude" type="t:PreludeT">
  <part name="POentries" type="t:acclInfoT">
  
```

Provide a Port/OP. names, to be referenced by Binding

Provide a name for that message in this context, to be referenced by Binding  
 Default message name – operation + request/response

Refer to a message definition using the WSDL's target namespace

w.xmlns=".../wsdl/..."

Whereas messages use schema namespace

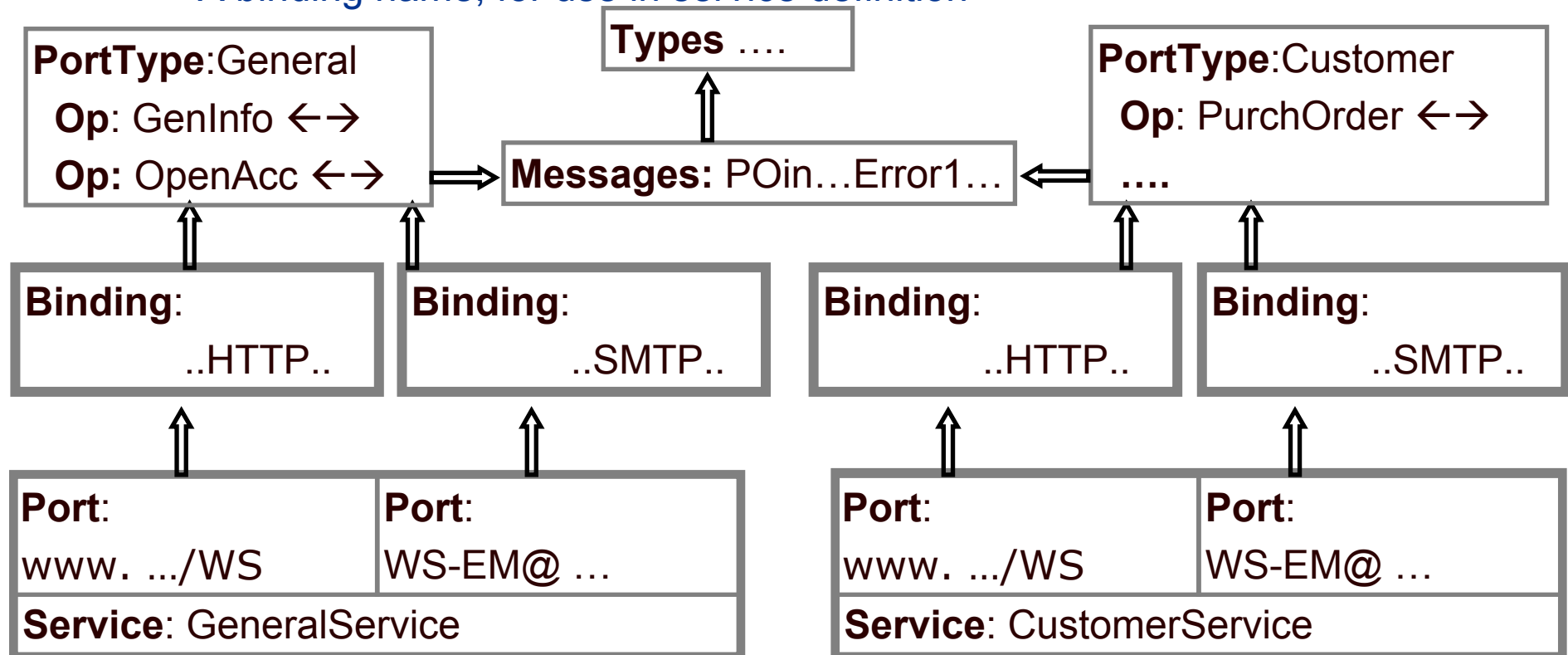
t.xmlns=".../types/..."

## Goals –

- To be able to understand
  - WSDL definition for a standard SOAP binding
  - A soap message
  
- Structure
  - SOAP Messages
  - General Structure of WSDL
  - Details of abstract Service Definition
  - Details of Physical Service Definition - Core
  - Physical Service Definition - Extensions

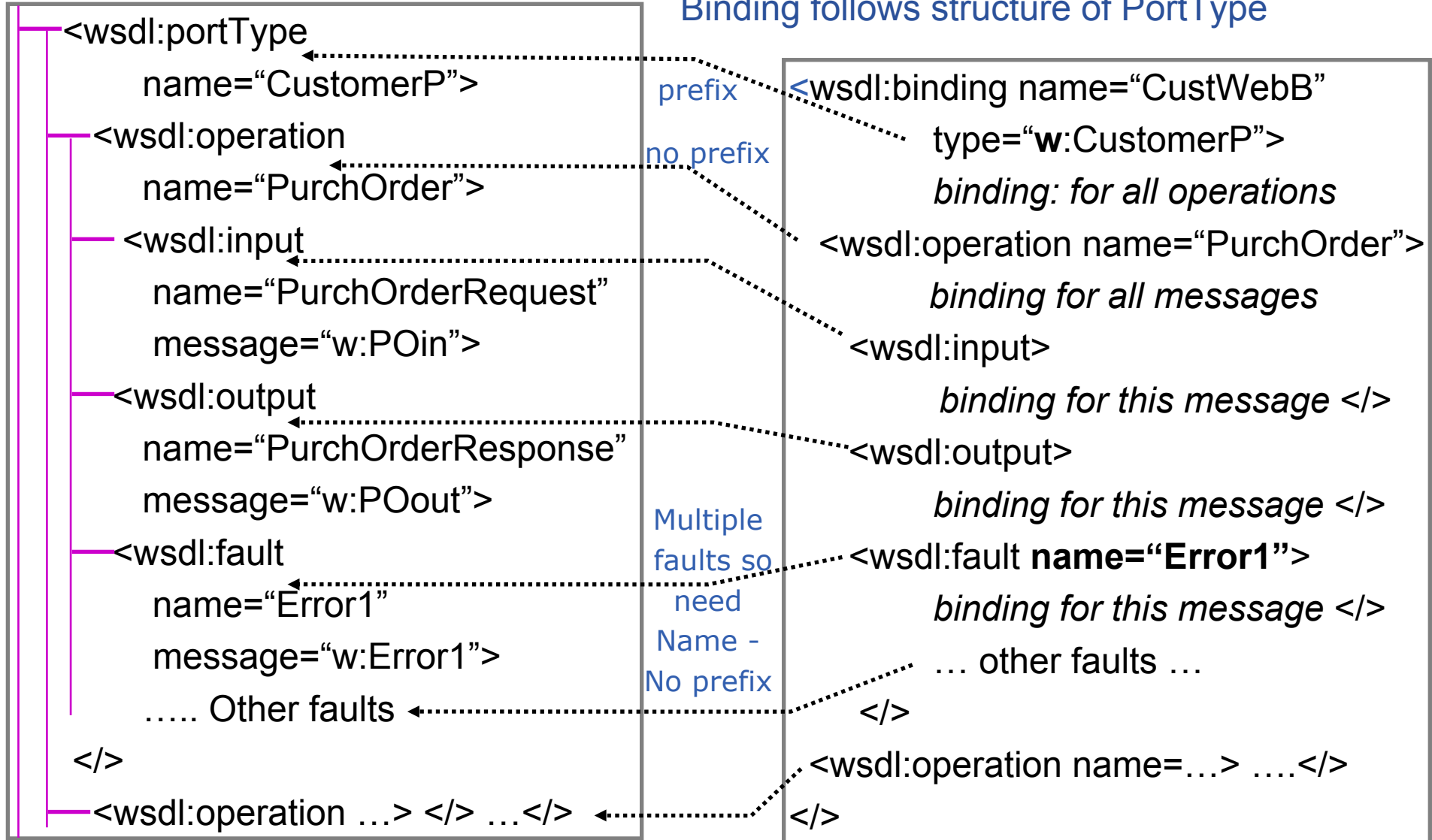


- A Binding defines
  - A particular PortType – named as its “type” – General or Customer
  - Particular message format and communication protocol details
    - By extensibility point
    - A standard extension is SOAP binding
  - A binding name, for use in service definition



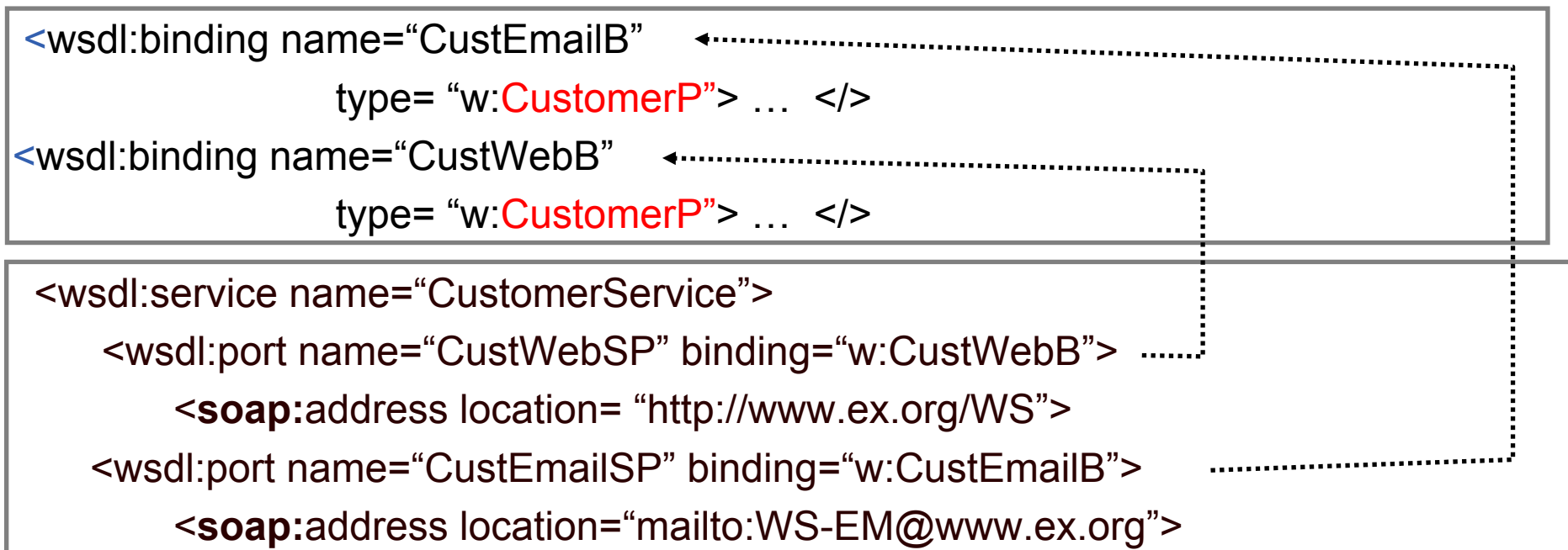
# Structure of Binding Element

Binding follows structure of PortType






- Can have multiple services in one WSDL definition document
- Each Service can have multiple ports, each bound to a binding
- For WSDL 2.0 – all ports of a service must have the same portType
  - Can have different portTypes in WSDL 1.1 –  
consumer may need all functionalities for the service to be useful
- Two ports having the same portType means same semantics
- Gives the location, a URL –
  - this is a SOAP extension of WSDL, not WSDL core



## Goals –

- To be able to understand
  - WSDL definition for a standard SOAP binding
  - A soap message
  
- Structure
  - SOAP Messages
  - General Structure of WSDL
  - Details of Abstract Service Definition
  - Details of Physical Service Definition - Core
  - Physical Service Definition - Extensions 

- There are a number of defined bindings to be used in the extension points
  - **SOAP** – identifying the **SOAP 1.1** standards
    - Transport
      - *Over HTTP*
      - *Over SMTP*
      - ....
    - Style
      - *RPC*
      - *Document*
    - Use
      - *Literal*
      - *Encoded*
  - HTTP
  - MIME
- **SOAP over HTTP** is most commonly used
  - all we will deal with here

# The Soap Binding Extension

```

<wsdl:binding name="CustWebB" type="w:CustomerP">
  <soap:binding style="document"
    transport="http://schemas.xmlsoap.org/soap/http" >
  <wsdl:operation name="PurchOrder">... </>
  <wsdl:operation name="...">...</> .....
</>

```

PORT LEVEL  
Replaces an ANY extension point in general definition

soap:binding element means Using SOAP standards. Message structure is

```

<soap:envelope>
  <soap:header>...</>
  <soap:body>...</>

```

style=... - Default for all operations;

- = "rpc" – body is parameters/return
- = "document" – body is one document

Optional  
default = "document"

Transport=  
URI to identify some protocol  
Optional

# The Soap Binding Extension

```

<wsdl:binding name="CustWebB" type="w:CustomerP">
  <soap:binding style="document"
    transport="http://schemas.xmlsoap.org/soap/http" >
  <wsdl:operation name="PurchOrder">
    <soap:operation
      soapAction="http://ex.org/PO"
      style="rpc">
    <wsdl:input> ... </> .... </>
  <wsdl:operation name="...">...</> .... </>

```

PORT LEVEL

OPERATION LEVEL

Replaces an ANY extension point in general definition

soapAction =  
 URI, the value for the HTTP header "SOAPaction"  
 Mandatory for SOAP/HTTP  
 For "document" style gives the operation  
 For JAX-RPC – empty, ""

style =  
 Over-rides port-level style

# The Soap Binding Extension

```

<wsdl:binding name="CustWebB" type="w:CustomerP">
  <soap:binding style="document"
    transport="http://schemas.xmlsoap.org/soap/http" >
  <wsdl:operation name="PurchOrder">
    <soap:operation
      soapAction="http://ex.org/PO"
      style="rpc">
    <wsdl:input>
      <soap:body use="encoded"
        encodingStyle="http://.../encoding/"
        namespace="http://ex.org/wsdl/Cust" /></>
      .... </>
    <wsdl:operation name="...">...</> .... </>
  
```

PORT LEVEL

OPERATION LEVEL

MESSAGE LEVEL

Namespace=

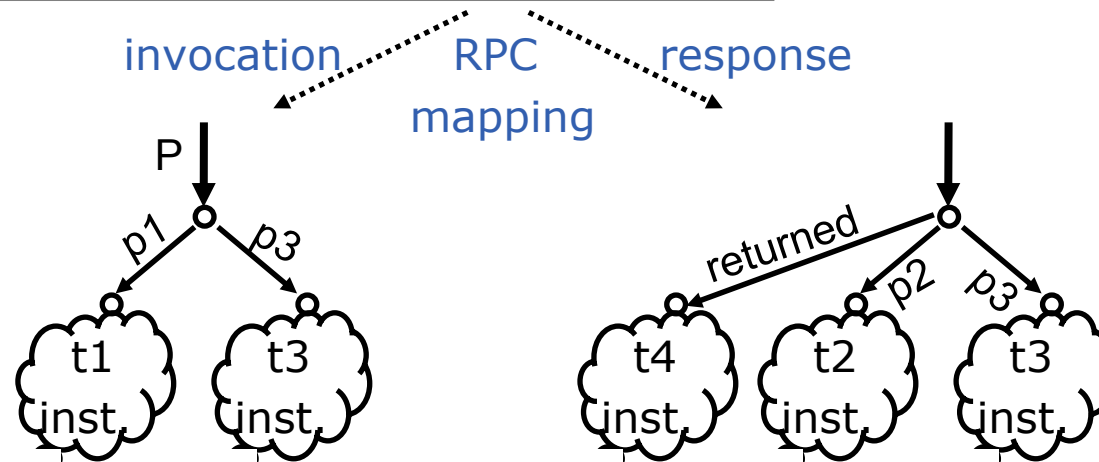
the namespace to be used for validating the outermost elements

Use = "Encoded" : each message part references a type – an abstract type encoded using the scheme(s) identified by encodingStyle

Use = "literal" : each message part ref's a type or element which gives the concrete format

encodingStyle = ".." a URI list to identify encoding scheme(s)  
 Soap-encoding is <http://schemas.xmlsoap.org/soap/encoding>

To.P( $\downarrow p1::t1$ ,  $\uparrow p2::t2$ ,  $\updownarrow p3::t3$ ):: t4



- **RPC**
  - Hint that this is best dealt with as a procedure call (/return)
  - Message parts are parameters which are wrapped as one component of Body
  - As in the SOAP RPC standard
- **Document**
  - This is a document to be processed – message parts are directly in body
  - Wrapped convention – single message part – looks like RPC style

```
<wsdl:message name="POin">
  <part name=prelude type=...>
  <part name=POentries type=...>
</>
```

```
<wsdl:message name="POout">
  <part name=Result type=...>
  <part name=delivSched type=...>
</>
```

```
<wsdl:operation name="PurchOrder">
  <wsdl:input name="PurchOrderRequest" message="w:POin">
  <wsdl:output name="PurchOrderResponse" message="w:POout">
```

```
<env:Body>
  <PurchOrderRequest>
    <Prelude> ... </>
    <POentries> ... </></></>
```



```
<env:Body>
  <PurchOrderResponse>
    <Result> ... </>
    <delivSched> ... </></></>
```

RPC  
Actual  
messages

```
<env:Body>
  <Prelude> ... </>
  <POentries> ... </></></>
```



```
<env:Body>
  <Result> ... </>
  <delivSched> ... </></></>
```

Document  
Actual  
messages



## WSDL

- **Defines abstract structure of service interactions**
  - Including logical content of messages exchanged
- **Defines binding – how the messages are carried and represented**
  - Standard binding is for SOAP over HTTP
    - Message is an XML document, with a particular structure
    - Using particular types
- **Defines Service as a number of ports, each being address and binding**
- **A site where you can obtain WSDL definitions of services and see what SOAP messages are produced - <http://xmethods.com/>**

*THE END*